Setting network and Linux services

Rafal Kapela

During this labaroatory you will setup the network on the embedded Linux platform, and learn the basics of network data exchange with embedded platforms

Note, that in order to follow instructions in this lab session you need to have prepared the following configuration:

- PC computer with Linux Kubuntu installed on it in the lab.
- Galileo2 development board.
- Arduino mezanine board.
- I2C display extension.
- Photoresistor extension.
- Potentiometer extension.
- An SD card reader for PC computer.
- Power supply.
- USB UART cable.
- An SD card prepared on the last laboratory session.

The setup is shown in the following picture.

1. Powerup and login to the board.
2. Install `vim` editor. You will need to install the ncurses dependency first (if it is not installed yet). Try:
   `whereisncurses`
   If this does not return a path to the ncurses.h file, you will need to do:

```
wget http://ftp.gnu.org/pub/gnu/ncurses/ncurses-5.7.tar.gz
tar -xvzf ncurses-5.7.tar.gz
cd ncurses-5.7
./configure --prefix=/usr
make
make install
```

Listing 1: bash script for instalation of vim dependencies

3. Vim is downloaded via Mercurial, which is not installed on the Galileo default, but you can use this git mirror site to clone it. More details here: https://github.com/b4winckler/vim

```
git clone https://github.com/b4winckler/vim.git
cd vim
./configure --with-features=huge
make
make install
```

Listing 2: bash script for vim instalation

4. Prepare network connection on your board:

   (a) create and open `init_net.sh` shell script:

   `vi init_net.sh`

   (b) put the following content inside the file:

```
#!/bin/bash
ifconfig <interface> down
ifconfig <interface> hw ether <X>
ifconfig <interface> up
ifconfig <interface> 150.254.21.<X>
ifconfig <interface> netmask 255.255.255.128
ifconfig <interface> broadcast 150.254.21.127
```

   Listing 3: bash script for network initialization

   In order to figure out the correct parameters of the script use `ifconfig` command. You need to unplug the neighbouring PC from the network and clone its MAC address on the Galileo.

   (c) Initialize network by running the script:

   `chmod +x init_net.sh`
   `./init_net.sh`

   (d) Check the connectivity with client-server application:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
```

```c
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#include <netdb.h>

void error(const char *msg) {
    perror(msg);
    exit(0);
}

int main(int argc, char *argv[]) {
    int sockfd, portno, n;
    struct sockaddr_in serv_addr;
    //struct hostent *server;

    char buffer[256];

    portno = 2425;
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd < 0) {
        error("ERROR opening socket");
    }

    bzero((char *) &serv_addr, sizeof(serv_addr));
    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(portno);
    serv_addr.sin_addr.s_addr = inet_addr("
        150.254.21.85");
    if (connect(sockfd,(struct sockaddr *) &serv_addr,
        sizeof(serv_addr)) < 0) {
        error("ERROR connecting");
    }

    printf("Please enter the message: ");
    bzero(buffer,256);
    fgets(buffer,255,stdin);
    n = write(sockfd,buffer,strlen(buffer));
    if (n < 0) {
         error("ERROR writing to socket");
    }

    bzero(buffer,256);
    n = read(sockfd,buffer,255);
    if (n < 0) {
         error("ERROR reading from socket");
    }

    printf("%s\n",buffer);
    close(sockfd);
    return 0;
}
```

Listing 4: client code

```c
#include <stdio.h>
#include <stdlib.h>
```

```c
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

void error(const char *msg) {
    perror(msg);
    exit(1);
}

int main(int argc, char *argv[])  {
     int sockfd, newsockfd, portno;
     socklen_t clilen;
     char buffer[256];
     struct sockaddr_in serv_addr, cli_addr;
     int n;
     sockfd = socket(AF_INET, SOCK_STREAM, 0);
     if (sockfd < 0) {
        error("ERROR opening socket");
     }

     bzero((char *) &serv_addr, sizeof(serv_addr));
     portno = 2425;
     serv_addr.sin_family = AF_INET;
     serv_addr.sin_addr.s_addr = INADDR_ANY;
     serv_addr.sin_port = htons(portno);
     if (bind(sockfd, (struct sockaddr *) &serv_addr,
         sizeof(serv_addr)) < 0) {
       error("ERROR on binding");
     }

     listen(sockfd,5);
     clilen = sizeof(cli_addr);
     newsockfd = accept(sockfd,(struct sockaddr *) &
         cli_addr,&clilen);
     if (newsockfd < 0) {
          error("ERROR on accept");
     }

     bzero(buffer,256);
     n = read(newsockfd,buffer,255);
     if (n < 0) error("ERROR reading from socket");

     printf("Here is the message: %s\n",buffer);
     n = write(newsockfd,"I got your message",18);
     if (n < 0) error("ERROR writing to socket");

     close(newsockfd);
     close(sockfd);
     return 0;
}
```

Listing 5: server code

5. Write an application that check for a file and if it exists sends it to the client after successful client-server connection.